

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

---

In re Letters Patent of:  
Kumar, Nishit et al.

Patent No.: 7,334,132

Issued: February 19, 2008

For: FLEXIBLE AND SCALABLE  
ARCHITECTURE FOR TRANSPORT  
PROCESSING

---

**REQUEST FOR CERTIFICATE OF CORRECTION  
PURSUANT TO 37 CFR 1.323 AND 1.322**

Attention: Certificate of Correction Branch  
Commissioner for Patents  
P.O. Box 1450  
Alexandria, VA 22313-1450

Dear Sir:

Upon reviewing the above-identified patent, Patentee noted typographical errors which should be corrected.

The typographical errors marked with an "A" on the attached list are found in the application as filed by applicant. Payment in the amount of \$100.00 covering the fee set forth in 1.20(a) is enclosed.

The typographical errors marked with a "P" on the attached list are not in the application as filed by applicant. Also given on the attached list are the documents from the file history of the subject patent where the correct data can be found.


The errors now sought to be corrected are inadvertent typographical errors the correction of which does not involve new matter or require reexamination.

Transmitted herewith is a proposed Certificate of Correction effecting such corrections.  
Patentee respectfully solicits the granting of the requested Certificate of Correction.

The Commissioner is authorized to charge any deficiency of up to \$300.00 or credit any excess in this fee to Deposit Account No. 04-0100.

Dated: April 1, 2008

Respectfully submitted,

By  \_\_\_\_\_  
Flynn Barrison

Registration No.: 53,970  
DARBY & DARBY P.C.  
P.O. Box 770  
Church Street Station  
New York, New York 10008-0770  
(212) 527-7700  
(212) 527-7701 (Fax)  
Attorneys/Agents For Applicant

## UNITED STATES PATENT AND TRADEMARK OFFICE CERTIFICATE OF CORRECTION

PATENT NO. : 7,334,132

Page 1 of 1

APPLICATION NO.: 10/608,301

ISSUE DATE : February 19, 2008

INVENTOR(S) : Kumar et al.

It is certified that an error appears or errors appear in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

On Sheet 12 of 19, in Fig. 12B, line 2, delete "SignAl" and insert -- Signal --, therefor.

On Sheet 12 of 19, in Fig. 12B, line 7, delete "SignAl" and insert -- Signal --, therefor.

In column 2, line 25, after "module" insert -- . --.

In column 2, line 28, after "invention" insert -- . --.

In column 7, line 11, before "data" delete "(".

In column 12, line 34, delete "11360," and insert -- 1360, --, therefor.

In column 19, line 55, in Claim 19, delete "demultiplexer" and insert -- demultiplexer --, therefor.

In column 20, line 57, in Claim 33, delete "complete" and insert -- complete --, therefor.

In column 21, line 20, in Claim 36, delete "demultiplexer" and insert -- demultiplexer --, therefor.

### MAILING ADDRESS OF SENDER (Please do not use customer number below):

John W. Branch, Esq.  
DARBY & DARBY P.C.  
P.O. Box 770  
Church Street Station  
New York, New York 10008-0770

1

This collection of information is required by 37 CFR 1.322, 1.323, and 1.324. The information is required to obtain or retain a benefit by the public which is to file (and by the USPTO to process) an application. Confidentiality is governed by 35 U.S.C. 122 and 37 CFR 1.14. This collection is estimated to take 1.0 hour to complete, including gathering, preparing, and submitting the completed application form to the USPTO. Time will vary depending upon the individual case. Any comments on the amount of time you require to complete this form and/or suggestions for reducing this burden, should be sent to the Chief Information Officer, U.S. Patent and Trademark Office, U.S. Department of Commerce, P.O. Box 1450, Alexandria, VA 22313-1450. DO NOT SEND FEES OR COMPLETED FORMS TO THIS ADDRESS. **SEND TO: Attention Certificate of Corrections Branch, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.**

# Darby & Darby

Issued Patent Proofing Form  
Note: P = PTO Error

File#: 21333/0209054-USO

A = Applicant Error

US Serial No.: 10/608,301

US Patent No.: US 7,334,132 B1

Issue Dt.: Feb. 19, 2008

Title: **FLEXIBLE AND SCALABLE ARCHITECTURE FOR TRANSPORT PROCESSING**

Sr. No.	P/A	Original		Issued Patent		Description Of Error
		Page	Line	Column	Line	
1	A	Sheet 12 of 19 Drawings-only black and white line drawings (12/18/2007)	2 (Fig. 12B)	Sheet 12 of 19 (Fig. 12B)	2	Delete "SignAl" and insert - - Signal - -, therefor.
2	A	Sheet 12 of 19 Drawings-only black and white line drawings (12/18/2007)	7 (Fig. 12B)	Sheet 12 of 19 (Fig. 12B)	7	Delete "SignAl" and insert - - Signal - -, therefor.
3	A	Page 5 Specification (06/27/2003)	16	2	25	After "module" insert - - . - .
4	A	Page 5 Specification (06/27/2003)	18	2	28	After "invention" insert - - . - .
5	P	Page 16 Specification (06/27/2003)	11	7	11	Before "data" delete "(".
6	P	Page 27 Specification (06/27/2003)	13	12	34	Delete "11360," and insert - - 1360, - -, therefor.
7	A	Page 8 Claims (12/18/2007)	Claim 33 Line 3	19	55	In Claim 19, delete "demultiplexer" and insert - - demultiplexer - -, therefor.
8	A	Page 8 Claims (12/18/2007)	Claim 30 Line 3	20	57	In Claim 33, delete "compete" and insert - - complete - -, therefor.
9	A	Page 8 Claims (12/18/2007)	Claim 33 Line 3	21	20	In Claim 36, delete "demultiplexer" and insert - - demultiplexer - -, therefor.

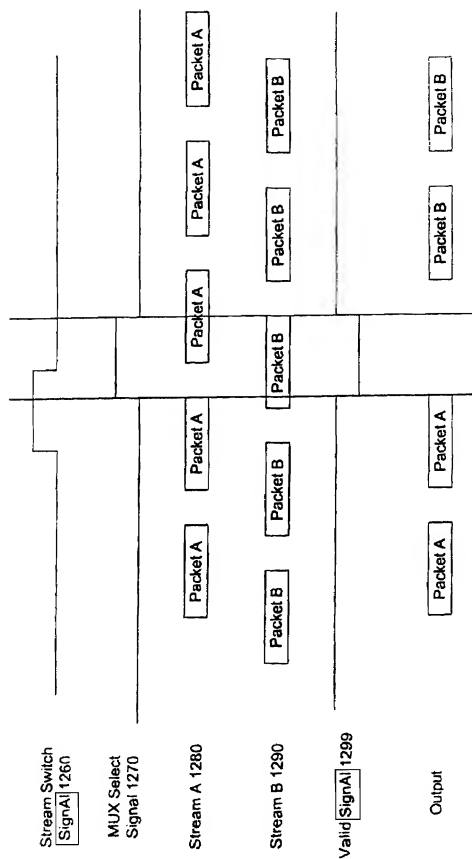


Fig. 12B

# FLEXIBLE AND SCALABLE ARCHITECTURE FOR TRANSPORT PROCESSING

## FIELD OF THE INVENTION

This invention relates to digital audio and video processing and particularly concerns system and methods for performing transport related functions in a digital video receiver.

## BACKGROUND

Digital video is being used in an increasing array of applications ranging from personal computers (PC) and videoconferences to digital televisions (TVs), set-top boxes, and personal video recorders (PVR). These varied video processing systems have a myriad of content delivery types, ranging from cable, satellite, and terrestrial broadcasts, to streaming video and video-on-demand over the Internet. Despite attempts to standardize, these delivery infrastructures (e.g., DirecTV, ATSC, DVB, US Cable, and ARIB) vary in their formats, conditional access standards, and demultiplexing requirements. Furthermore, these video products are getting more-and-more interactive and sophisticated, and are evolving in the features and services they support.

Demands of high performance and flexible systems with capabilities to address such varying standards and formats and evolving features have created many design challenges. Therefore, there is a need in the technology to develop a flexible and scalable Transport Processor architecture that addresses these varied applications, different content delivery systems, varied formats and standards, and evolving features and requirements.

## SUMMARY OF THE INVENTION

A method and apparatus for a multi-stream transport architecture is described.

## BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example, and not by way of limitation, in the Figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

FIG. 1 is a diagram of one embodiment of a complete system-on-a-chip in which the Transport Processor may be used.

FIG. 2 is a diagram illustrating one embodiment of the primary data-flow of signals received into the system-on-a-chip up to displaying video on a TV/VCR and outputting audio to speakers.

FIG. 3 is a diagram illustrating one embodiment of the data-flow and control-flow within the system-on-a-chip to provide conditional access.

FIG. 4A is a diagram illustrating one embodiment of the Transport Processor according to the present invention.

FIG. 4B is a diagram illustrating the first-level micro-architecture of one embodiment of the Transport Processor according to the invention.

FIG. 5 is a diagram illustrating one embodiment of the data-flow within the Transport Processor to provide a "Watch-a-Program" application.

FIG. 6 is a diagram illustrating one embodiment of the data-flow within the Transport Processor to provide a "Delayed Playback" Application.

FIG. 7 is a diagram showing one embodiment of a proprietary transport packet format.

FIG. 8 is a diagram illustrating one embodiment of the Front-End module.

FIG. 9 is a diagram illustrating one embodiment of the external input-output circuitry, which is part of the Front End module.

FIG. 10 is a diagram illustrating one embodiment of the external input circuitry.

FIG. 11 is a diagram illustrating one embodiment of the external output circuitry.

FIG. 12A is a diagram illustrating one embodiment of the switching matrix, which is part of the Front-End module.

FIG. 12B is an exemplary data diagram, illustrating the packet switching by the switching matrix.

FIG. 13 is a diagram illustrating one embodiment of the packet processor, which is part of the Front-End module.

FIG. 14 is a diagram illustrating the PID filter, which is part of the Front-End module, according to one embodiment of the invention.

FIG. 15 is a diagram illustrating one embodiment of the modes of operation for the Playback module.

FIG. 16 is a diagram illustrating the Playback module, which is part of the Transport Processor, according to one embodiment of the invention.

FIG. 17 is a diagram illustrating one embodiment of the Descrambler module, which is part of the Transport Processor.

FIG. 18 is a diagram containing three tables illustrating the Descrambler throughputs according to the implementation of one embodiment of the invention.

FIG. 19 is a diagram illustrating one embodiment of the Transport Demultiplexer module, which is part of the Transport Processor.

FIG. 20 is a diagram illustrating one embodiment of the size optimization of the Data RAM in the Transport Demultiplexer.

## DETAILED DESCRIPTION

A method and apparatus to perform transport related functions on incoming transport stream(s) from either tuner chip(s) (for broadcast content) or from the memory (for pre-stored content) is described. A Front-End module receives a plurality of transport streams and provides the multiplexers to select the streams that are of interest. Using the PID (Packet ID) filter circuitry in the Front-End, only the packets of interest are retained. After being appended with relevant header and footer information, the relevant packets are written into a common memory buffer. This additional information appended to every packet is used to create an aggregate transport stream, so that a single instantiation of the processing units can handle the aggregate stream. In one embodiment, a proprietary 208-byte per packet format is used to ensure that there is no loss of information. The information appended to each packet, includes, for example, the arrival time of a packet and the transport stream that a packet belongs to.

A Readback circuit reads the packets from the memory buffer one-by-one and sends the packets to the Descrambler circuit for conditional access functions. The descrambled packets are then passed to a flexible microcontroller-based Demultiplexer circuit to provide a myriad of transport demultiplexing functions. The outputs of the Demultiplexer

In one embodiment, the Demultiplexer 470 also contains circuitry to control the VCXO (Pulse-Width Modulated) Control signal. The VCXO Control signal fine-tunes the frequency of the reference clock generated by the external crystal, and is used to prevent long-term drifts in playing out video. The goal of fine-tuning is to match video display rate in the SOC (decoder side) with the intended rate (encoder side). In one embodiment the Transport Processor provides circuitry required to control two independent VCXOs. This is used for dual high-definition decode and display applications, where the transport data for the two displays arrive into the SOC through either two channels from the same digital receiver or from two different digital tuners.

The Host-Bus Interface unit 480 provides a common register-programming interface for blocks within the Transport Processor 120. In one embodiment, the Host Bus Interface unit 480 physically contains the storage for programming registers and contains the address decode circuitry for registers as well as internal RAM. It also contains the necessary circuitry to generate interrupts.

The Memory-Bus Interface 440 provides a common memory read-write interface to blocks within the Transport Processor 120. The Memory-Bus Interface 440 provides arbitration logic for the internal blocks to initiate data transfer, on the inside. In one embodiment, the Memory-Bus Interface 440 arbitrates data transfers on the Internal Memory Bus of the SOC (not shown), between Front-End (Writes), Playback (Reads), Readback (Reads), and Transport Demultiplexer (Writes). The Memory-Bus Interface 440 toggles signals relevant to the Internal Memory Bus protocol, on the outside.

FIG. 5 shows the dataflow within Transport Processor 120 for a simple "WATCH-A-PROGRAM" application. Digital transport stream 510 is received by the Front-End 410 from a Digital Receiver. Selected transport packets after PID filtering (within Front-End) are appended with their respective arrival times and the stream ID, and are written 520 to a circular buffer in the memory by the Front-End 410 through the Memory-Bus Interface 440. The Readback 450 reads out packets 530 one-by-one from this circular buffer. Each packet is descrambled by the Descrambler 460.

The Transport Demultiplexer 470 then parses descrambled packets and separates out video data, audio data, and control information. The Demultiplexer 470 then writes 560 demultiplexed outputs into separate circular buffers through the Memory-Bus Interface 440. The compressed video data buffer is read out by Digital Decoder (not shown) for MPEG decode and subsequent display. Likewise, the audio buffer is read by Audio Processor (not shown) for decompression and final audio output.

FIG. 6 shows dataflow for a more complex "DELAYED PLAYBACK" application. This is a common application in PVR systems, where a user can pause live broadcast and watch the broadcast after a fixed delay. In this case the transport stream from the Digital Receiver is handled the same way as the "WATCH-A-PROGRAM". The only difference is that the Transport Demultiplexer 470, instead of parsing and binning the transport packets, simply writes out a transport stream into the Memory through Memory Interface 130.

In one embodiment, the stream written out by the Transport Demultiplexer is a partial transport since the unwanted packets are dropped by the PID filter (not shown). In one embodiment, the Transport Demultiplexer 470, in a PVR application, also writes out a side channel with a table of I-pictures (MPEG standard Intra-coded pictures). This side-channel information is used for random access and naviga-

tion of content at a later time. In one embodiment, the partial transport stream is moved from Memory to a hard disk using the DMA (Direct Memory Access) engine in Super IO 190. The data path for the playback video originates from the hard disk 193. In one embodiment, while the recording on the hard disk 193 is in progress, the Super IO 190 can read out previously stored content from the hard disk and send it to a circular buffer in the Memory 196. The data in the circular buffer is read by the Playback unit 430 of the Transport Processor 120. A partial transport stream, with gaps for the missing packets, is re-created by the Playback unit 430 and fed to the Front-End 410. Hence in this application the Front-End 410 writes out packets from two separate streams, although they are only temporally different, after tagging them with stream ID and arrival time stamp. Packets from both streams are written into a common circular buffer in Memory.

The Readback block 450 reads packets from this circular buffer. In one embodiment, only packets from the live stream are descrambled. For the delayed stream, all transport packets are bypassed within the Descrambler 460. The packets from delayed stream are parsed and binned appropriately for subsequent decoding (both audio and video) and display, on similar lines as the "WATCH-A-PROGRAM" application.

FIG. 7 shows one embodiment of the proprietary 208-byte format for packets. These packets are written, in one embodiment, by the Front-End into a memory buffer. The Transport Processor supports MPEG and DirecTV formats, and may support additional formats as well. For MPEG, a 188-byte transport packet is appended with a header and stuffing at the end. In one embodiment, the header is 8-bytes, and the Stuffing is the remaining 12-bytes. The header for MPEG contains a Stream ID and arrival timestamp. In one embodiment, the Stream ID is a 2-bit Stream ID (since a maximum of four streams are written in memory by the Front-End), and the time stamp is a 42-bit MPEG-compliant arrival timestamp. In one embodiment, the header further includes a 1-bit flag set to 0, and a 16-bit packet count. The 12-byte Stuffing has all reserved bytes except one that contains user-programmable bits. The user-programmable bits, for example, may be used for identifying the source of a transport stream (e.g., satellite tuner, terrestrial tuner, IEEE 1394).

The information appended in header and footer of a 208-byte packet are read and used by the Descrambler and the Transport Demultiplexer. In one embodiment, the arrival timestamp is used by the Transport Demultiplexer to perform a hysteresis between PCR timestamps and the arrival timestamp in order to control the VCXO Control signal for PCR locking. Likewise, in one embodiment the stream ID is used by the Descrambler to decide which set of keys should be used to descramble a particular packet.

For DirecTV, a 130-byte transport packet has the same 8-byte header but is stuffed with 70 bytes instead. Everything is similar to MPEG, except that the timestamp field is only (DirecTV compliant) 32 bits wide. It must be emphasized that the Transport Processor architecture is designed with enough flexibility to process any mix of MPEG and DirecTV streams, and the above process may be applied to any other type of stream.

The architecture is optimized such that a single instance of Descrambler and Transport Demultiplexer can handle an aggregate stream of packets that came into the Front-End as different streams. This optimization is possible in the domain of transport processing since a single transport stream is no more than 10 Megabytes/second. On the other hand cost-efficient designs of the Descrambler and Transport

11

internal core-clock domain. From this circuit onwards, the Transport Processor is clocked by the core clock. In one embodiment, a "clean" transport stream in the transport clock domain is also retained, to be used to route streams out of the Transport Processor through the four External Outputs.

The External Input 920 outputs two streams, Stream A and Stream B. If the External Input 920 is configured as a parallel port, both Stream A and Stream B contain the same incoming parallel transport stream (CLOCK, FRAME, VALID, and DATA [7:0]). If the External Input 920 is configured as a pair of serial ports, Stream A represents the serial transport stream from the 4 pins CLOCK, FRAME, VALID, DATA [0], and Stream B represents the serial transport stream from the 4 pins DATA [4] (for CLOCK), DATA [5] (for FRAME), DATA [6] (for VALID), and DATA [7] (for DATA). In this mode of operation the pins DATA [1], DATA [2] and DATA [3] are unused.

On similar lines as External Input 920, the External Output 930 consists of a pair of Serial Output Blocks 1110, 1150 and a Parallel Output Block 1160, as shown in FIG. 11. The Parallel Output Block 1160 contains simple multiplexers 1170 to choose a user-programmable transport stream to be routed out on the parallel port to a Digital Recorder. As discussed above, all 8 input transport streams (8=2 Streams each from the other 3 ports+PlayBack+NRSS) that are inputs to the multiplexer 1070 are already in a normalized parallel transport format; since each serial stream first goes through a Serial to Parallel Converter in External Input.

A Serial Output Block 1110 is however more involved. A user can not only choose transport data source from one of the nine transport streams (9=2 Streams each from the other 3 ports+1 Stream from the other 4 pins on the same port+PlayBack+NRSS) but also choose an independent source for bit clock. In one embodiment, the multiplexer 1130, 1140 inputs for clock source contains all the 7 incoming clocks and the PlayBack clock, besides clocks from the Serial Clock Generators (shown in FIG. 9). An asynchronous FIFO 1115 is employed to resample the selected transport stream with an independently chosen FIFO Read clock. The FIFO Read Clock has to be at least eight times faster than the FIFO Write clock (Parallel clock). One of the Serial Output Blocks 1110 outputs Serial Transport Stream A, while the other Serial Output Block 1150 outputs Serial Transport Stream B. Multiplexer 1180 selects whether to output the two serial streams or the one parallel stream.

The Switching Matrix, as shown in FIG. 12A, contains four sets of multiplexers 1220, 1230, 1240, 1250 to select the four streams to be captured in memory. The Switching Matrix 820 supports dynamic changing of multiplexer selects. That is, the entire Front-End need not be brought down in order to accomplish a switching action from one stream to another. This is used, for example, in record-and-watch applications where starting a program recording should not introduce any glitches in the transport stream (or program) being watched. The support for dynamic switching is provided by the Stream Select Delay Unit 1210, which simply delays the multiplexer select signals to make a switch only on proper packet boundaries.

FIG. 12B illustrates one embodiment of the signals. The stream selection is done by a user-programmed external (register) select 1260, which controls an internal select 1270 of the multiplexer. In one embodiment, the internal select 1270 to the multiplexer is activated only after the packet-end signal of the original stream 1280 is seen. Thus, the system does not switch to a new stream 1290 until a packet-end signal of the current packet in the original stream 1280 has

12

been received, so that only complete packets are propagated. In order to ensure that only complete packets are propagated on the new stream 1290 as well, the data valid control 1299 is used. The data valid signal 1299 is changed to the new stream (indicating that the data in the new stream is valid) only after a packet end signal is received on the new stream 1290. This ensures that only complete packets are propagated, from both the original and the new stream.

The Packet Processor, as shown in FIG. 13, contains four sets of circuitry to capture packets for four streams and append header and footer information. In one embodiment, the Packet Processor 830 creates a 208-byte proprietary packet format. Incoming bytes for a stream are first garnered into 64-bit words using the Shift Registers 1310, 1330, 1350, 1370. Contents of the Shift Registers 1310, 1330, 1350, 1370 are moved to their respective Shifter FIFOs 1320, 1340, 1360, 1380. In one embodiment, the Shifter FIFOs 1320, 1340, 1360, 1380 are 5 words deep. From the Shifter FIFOs 1320, 1340, 1360, 1380, the data is written to a Front-End Packet RAM 1395, common among all four streams.

The Shifter FIFOs 1320, 1340, 1360, 1380 hold packet data until the PID value for the ongoing packet is either matched or rejected by the PID Filter. Since the PID Filter requires a 64-cycle delay in the worst case and it is shared among all the four streams, packet data in the Shifter FIFOs 1320, 1340, 1360, 1380 may be held for several cycles. If the PID Filter finds a match for a given packet, all bytes of data in the Shifter FIFO relevant to that packet are written out to the Packet RAM 1395 through the RAM Arbitrator 1390. If the PID filter rejects the PID value, all data for that packet arriving in the Shifter FIFO is thrown away.

The other benefit provided by the Shifter FIFOs 1320, 1340, 1360, 1380 is handling the worst-case arbitration delay. In one embodiment, the 208-word Front-End Packet RAM 1395 is implemented as a single-ported RAM, with eight write clients and a bursty read client. The read client issues requests in large bursts in back-to-back cycles to efficiently transfer complete data packets to the memory through the Memory Bus Interface. Control signals are transferred through requestor 1399 to the Memory Bus Interface. In the arbitration scheme, in one embodiment, the read client has the highest priority, and the other eight clients are arbitrated in a round-robin fashion. In one embodiment, the contents of the Packet RAM 1395 are transferred out to Memory in two separate bursts (of either 10 and 16, 12 and 14, 14 and 12, or 16 and 10; depending on the alignment of the Memory write address within a 16 word page) after all the 208 bytes (208 bytes=26 words) of a packet have been completely written into the Packet RAM 1395.

In one embodiment, the Front-End Packet RAM 1395 has space to store 2 packets for each stream. In one embodiment, only complete packets are written to the Memory since the input stream could stop anywhere within a packet due to various error conditions, and it is important not to propagate incomplete packets through the remainder of the Transport Processing pipeline (that is, Readback, Descrambler, and Transport Demultiplexer). The Packet Processor 830, in one embodiment, also contains small 4x9-bit FIFOs 1325, 1345, 1365, 1385 to store Stream Format and User-Byte information corresponding to a packet. In one embodiment, these are inserted in the header and footer, respectively, of a 208-byte packet.

FIG. 14 illustrates one embodiment of the PID Filter. In one embodiment, the PID Filter 850 is capable of handling four streams simultaneously. It accepts four PID Values, the corresponding Stream Format—that is, whether it is a 13-bit



19

8. The transport processor of claim 7, wherein the single format is a 208-byte format, and packets with fewer than 208 bytes are padded to comply with the single format.

9. The transport processor of claim 7, wherein the single format includes originating stream information that comprises temporal information.

10. The transport processor of claim 7, wherein the single format includes originating stream information that comprises stream identifier and additional user specified information.

11. The transport processor of claim 7, wherein the single format includes originating stream information.

12. The transport processor of claim 1, wherein the aggregate stream includes transport data obtained from different transport protocol standards.

13. The transport processor of claim 1, further comprising:

- a plurality of input/output (I/O) ports;
- an I/O port that is user-selectable to a parallel or serial format.

14. The transport processor of claim 13, wherein the I/O ports comprise:

- a serial output block to resample parallel data, and to convert the parallel data to serial data with an independently programmable bit clock selection.

15. The transport processor of claim 1, further comprising:

- a PID filter to discard packets from the aggregate transport stream, retaining only packets of interest;
- a descrambler to descramble the remaining packets in the aggregate stream; and
- a demultiplexer to demultiplex the descrambled packets in the aggregate stream;

wherein the descrambler and the demultiplexer receive only the packets of interest.

16. The transport processor of claim 1, further comprising:

- a switching matrix to select a subset of the streams out of a plurality of streams for storage and subsequent descrambling and demultiplexing.

17. The transport processor of claim 16, wherein the switching matrix comprises:

- a delay circuit to switch to a new stream after receiving an end of packet signal from an original stream, such that only complete packets from the original stream are propagated.

18. The transport processor of claim 17, wherein the switching matrix further comprises:

- a data valid signal to indicate that the output of the switching matrix is valid only after an end of packet signal is received from the new stream, such that only complete packets from the new stream are propagated.

19. The transport processor of claim 1, further comprising:

- a demultiplexer to demultiplex an aggregate transport stream retrieved from the memory, into a plurality of different streams, for use by said decode and display operations.

20. The transport processor of claim 19, further comprising a descrambler to receive the aggregate transport stream from the memory via the memory interface and to provide a descrambled version of the aggregate transport stream to the demultiplexer.

21. A system on a chip (SOC) comprising:  
a transport processor to PID filter, descramble, and demultiplex a plurality of transport streams, the transport processor including:

20

- a front end to concurrently receive a plurality of transport streams, where two or more of the transport streams have different formats, and

- a packet processor to create an aggregate transport stream having a single format from the plurality of transport streams;

- a memory to store demultiplexed outputs of the plurality of transport streams;

- and an output processor to retrieve one or more demultiplexed outputs from the memory and perform audio/video decode and display functions simultaneously.

22. The SOC of claim 21, wherein the output processor is a combination of digital audio decoder, digital video decoder, audio processor, and display processor.

23. The SOC of claim 22, wherein the audio and video frames for two independent transport streams are rendered without repeated or skipped frames.

24. The SOC of claim 21, wherein the transport processor further comprises:

- a readback logic to read packets from the memory, for descrambling and demultiplexing functions.

25. The SOC of claim 21, further comprising a memory interface for use by the transport processor and the output processor to access contents of the memory.

26. A front end in a transport processor to receive a plurality of transport streams from digital receivers, comprising:

- a switching matrix to receive concurrently the plurality of transport streams, where two or more of the plurality of transport streams have different formats, and to output a programmable subset of the plurality of transport streams;

- a packet processor to receive the subset of the plurality of transport streams and to aggregate the subset of the plurality of streams into a single aggregate transport stream in a single format.

27. The front end of claim 26, further comprising:

- a memory to store the aggregate transport stream.

28. The front end of claim 27, further comprising:

- a PID filter to discard packets, retaining only packets of interest.

29. The front end of claim 27, further comprising:  
an external input/output (I/O) to receive the plurality of transport streams, the external I/O having a plurality of bi-directional ports.

30. The front end of claim 29, wherein each of the bi-directional ports can be configured as either a single parallel or a pair of serial ports.

31. The front end of claim 30, wherein a bi-directional port includes a serial input block to receive serial input and generate a synchronized parallel output.

32. The front end of claim 30, wherein a bi-directional port includes a serial output block to generate a serial transport stream with an independent bit clock for output.

33. The front end of claim 26, wherein the switching matrix comprises:

- a stream select delay unit to ensure that only complete packets are propagated.

34. The front end of claim 26, wherein the packet processor is further to attach appropriate header and footer information to transport packets in the subset of the plurality of transport streams.

35. The packet processor in the front end of claim 34, wherein the packet processor generates packets of a uniform size, regardless of originating protocol.

36. A digital audio/video receiver system comprising,  
on a single chip:

21

a transport processor including:

a front end, the front end including:

a switching matrix to receive concurrently a plurality of transport streams, including video and audio, each including a plurality of packets, 5 where two or more of the plurality of transport streams have different media formats, each transport stream including a plurality of packets,

a PID filter to filter out packets that do not meet 10 specified criteria, and

a packet processor to create an aggregate transport stream in a single format from the plurality of transport streams;

a memory interface through which the transport 15 processor can store the aggregate transport stream in a memory for subsequent processing;

a descrambler to descramble packets read from the memory, and

a demultiplexer to demultiplex packets read from the 20 memory, for use by subsequent decode and display operations;

a digital decoder to perform video processing functions including decompression of video received from the memory and to store processed video in the memory; 25

an audio processor to perform audio processing functions including audio decompression on audio received from

22

the memory and to generate an audio output of said digital audio/video receiver system;

a graphics processor to process graphics;

and a display processor to produce a display output of said digital audio/video receiver system by combining processed graphics and video from a plurality of sources to generate a display in any of a plurality of different display formats.

37. A method comprising:

receiving concurrently, in a digital audio/video receiver system, a plurality of transport streams which have a plurality of different formats, each transport stream including a plurality of packets;

creating an aggregate transport stream in a single format from the plurality of transport streams in said digital audio/video receiver system; and

storing the aggregate transport stream in a memory for use by subsequent decode and display operations.

38. The method of claim 37, further comprising:

demultiplexing an aggregate transport stream retrieved from the memory into a plurality of different streams, for use by said decode and display operations.

39. The method of claim 38, further comprising:

descrambling the aggregate transport stream retrieved from the memory.

\* \* \* \* \*